2009

Computation of the Safety Fans for Multistage Aerodelivery Systems

Corley, M.S.

Monterey, California: Naval Postgraduate School

þÿ[Safety Fans GUI] Corley, M.S., and Yakimenko, O.A., Computation of th

20th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar
4 - 7 May 2009, Seattle, Washington

AIAA 2009-3003

Computation of the Safety Fans for Multistage Aerodelivery Systems

Melissa S. Corley,* and Oleg A. Yakimenko†
Naval Postgraduate School, Monterey, CA 93943-5107

This paper deals with the development of a mathematical foundation and practical algorithms to compute the safety fans to be used in conjunction with aerodynamic deceleration payload delivery systems tests. The paper provides formulas to compute a descent / gliding trajectory for a multiple-stage system and then considers several cases of major failures. It also introduces the concept of ballistic winds, simplifying calculation of the safety fans so that the computer is no longer needed, and shows how to carefully calculate these ballistic winds based on the best known winds aloft. Finally, the paper introduces the graphical user interface developed and tested by the authors that is supposed to provide test planning officers with a unique variety of trade-off options as well as to effectively represent all safety-related information both to the ground personal and to the crew performing a payload delivery from a transport aircraft or helicopter. The paper ends with conclusions.

Notations

H_0, V_0, Ψ_0	= altitude, speed and direction of the aircraft at the release point
H_2, H_3	= altitude AGL, at which the 2nd and 3rd stages are deployed (if present)
ΔH	= altitude loss before the 1st stage canopy deployment
x, y, h	= system's Northing, Easting and altitude
x_i, y_i	= Northing and Easting at the altitude of the i-th stage deployment (i=1,2,3)
$w(h), \psi_w(h)$	= speed and direction of the wind at altitude h
v_{vi}, GR_i	= descent rate and glide ratio during the i-th stage (i=1,2,3)

I. Introduction

AS with any other technical system, aerodynamic delivery systems (ADS) are subject to failures. That is why numerous tests are required to tune the system, make it more reliable and therefore minimize the risk of failure. A variety of ADSs rely on a parachute, a cluster of parachutes, or most recently autonomous parafoils to deliver payloads of different weight m (up to several tons) from different altitudes H_0 (up to several kilometers). The canopy opening failures result in a ballistic trajectory that transforms the initial kinetic ($0.5mV_0^2$) and potential (mgH_0) energy into kinetic energy at impact.

A failed, ballistically falling system can cause huge damage at impact. That is why it is important to foresee such an unfortunate event and take certain precautions organizing these drops in remote and uninhabited areas. Basic physics provides a very simple formula to calculate the ground distance the failed ADS can cover. If we neglect aerodynamic drag (yielding the largest ground distance estimate possible) this formula is as simple as V_0T, where the time of fall T can be estimated as $T = \sqrt{2H_0 g^{-1}}$. Of course if we have strong horizontal winds, they will definitely affect this range (horizontal projection of the trajectory will in this case not be a straight line any longer), though the effect will be small as the ADS would fall through these winds fairly quickly.

In case of partial canopy opening failures, however, the effect of winds will be stronger. This is not only because the system will descend slower and take longer to reach the ground, but also because we'll have to account for

* Ph.D. Student, Department of Mechanical and Aerospace Engineering, mscorley@nps.edu, Member AIAA.

† Research Associate Professor, Department of Mechanical and Astronautical Engineering, Code MAE/Yk, oayakime@nps.edu, Associate Fellow AIAA.

aerodynamic drag. In the case where we are dealing with a failed parafoil the situation can be even worse because of the high glide ratio of such systems. With a fully deployed canopy but with the controls stuck in some predetermined position or in the case of a controls failure, the parafoil-based ADS can glide far away from the intended point of impact (IPI).

This paper develops a mathematical foundation and corresponding software for reliable computation of safety fans for multistage systems (when the deployment of multiple canopies is staged) and is organized as follows. Section II provides the mathematical foundation for computing safety fans for multistage systems in the different types of failure modes described above (canopy opening failure and controls failure for controllable ADSs). Next, Section III presents a detailed example of computing these safety fans for a two-stage system. Section IV introduces the concept of so-called ballistic winds, the instrument allowing major simplification of safety fan computation. This simplification provides a reliable method for a range safety officer to compute these fans even without a computer by using a simple look-up table prepared upfront using the best known winds at the drop zone (DZ). Section V presents a graphical user interface (GUI) developed in the MATLAB development environment that provides much more flexibility in computing safety fans, suggesting release points and directions, and therefore allows for assessing and mitigating risks. The paper ends with conclusions.

II. Computation of Safety Fans

This section considers two major failures. The first case is when the very first stage canopy (or a single canopy in a one-stage system) doesn't open at all and the ADS falls ballistically. This will be the canopy opening failure case. The second case is when the first stage is deployed successfully but there are certain problems with steering the system in the desired direction. This case will be referred to as the controls failure case. The effect of winds in both cases will be accounted for as described above.

A. Canopy Opening Failure

Assuming that aerodynamic drag can be neglected, the equations driving the ADS motion are:

$$\begin{aligned} \frac{dx}{dt} &= w(h)\cos\psi_w(h) + V_0\cos\Psi_0 \\ \frac{dy}{dt} &= w(h)\sin\psi_w(h) + V_0\sin\Psi_0 \\ \frac{dh}{dt} &= gt \end{aligned} \tag{1}$$

Here the vertical motion is driven by the gravity force, while the horizontal motion is caused by the initial velocity and variable winds. Equations (1) can be further reduced to:

$$\begin{aligned} \frac{dx}{dh}\frac{dh}{dt} &= w(h)\cos\psi_w(h) + V_0\cos\Psi_0 \\ \frac{dy}{dh}\frac{dh}{dt} &= w(h)\sin\psi_w(h) + V_0\sin\Psi_0 \\ \frac{dh}{dt} &= gt \end{aligned} \tag{2}$$

From the last equation it follows that

$$t = \sqrt{\frac{2}{g}\left(H_0 - h\right)} \tag{3}$$

With this relationship between elapsed time and altitude the first two equations in (2) can be further rewritten as

$$\begin{aligned} \frac{dx}{dh} &= \frac{w(h)\cos\psi_w(h) + V_0\cos\Psi_0}{\sqrt{2g\left(H_0 - h\right)}} \\ \frac{dy}{dh} &= \frac{w(h)\sin\psi_w(h) + V_0\sin\Psi_0}{\sqrt{2g\left(H_0 - h\right)}} \end{aligned} \tag{4}$$

Now the *x* and *y* coordinates at any arbitrary altitude *h* can be obtained via integrating Eqs.(4):

$$x(h) = x_0 + \int_h^{H_0} \frac{w(h)\cos\psi_w(h) + V_0\cos\Psi_0}{\sqrt{2g\left(H_0 - h\right)}}\,dh$$
$$y(h) = y_0 + \int_h^{H_0} \frac{w(h)\sin\psi_w(h) + V_0\sin\Psi_0}{\sqrt{2g\left(H_0 - h\right)}}\,dh \tag{5}$$

The integrals in these equations can be further split into two separate components, so that

$$x(h) = x_0 + \int_h^{H_0} \frac{w(h)\cos\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{H_0 - h}{g}}V_0\cos\Psi_0$$
$$y(h) = y_0 + \int_h^{H_0} \frac{w(h)\sin\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{H_0 - h}{g}}V_0\sin\Psi_0 \tag{6}$$

If the parameters $w(h)$ and $\psi_w(h)$ are given as a look-up table with the elements: h_k, w_k and ψ_{wk}, $k = 1,...,N$, ($h_k > h_{k-1}$), then the effect of the winds can be reduced to the following finite sums:

$$\int_h^{H_0} \frac{w(h)\cos\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh \approx \sum_{k=m}^{M-1} \frac{w_{k+1}\cos\psi_{w;k+1}(h_{k+1} - h_k)}{\sqrt{2g\left(H_0 - h_k\right)}}$$
$$\int_h^{H_0} \frac{w(h)\sin\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh \approx \sum_{k=m}^{M-1} \frac{w_{k+1}\sin\psi_{w;k+1}(h_{k+1} - h_k)}{\sqrt{2g\left(H_0 - h_k\right)}} \tag{7}$$

(where H_0 corresponds to h_M and h to h_m).

Next, let us consider the case when the canopy opens successfully but the controls are stuck in some fixed position, causing the ADS to glide with a constant heading (ADS dynamics can be neglected).

B. Controls Failure

A multiple-stage ADS, say with three stages, assumes the following consecutive descent phases:

- Ballistic trajectory before the first stage is fully deployed;
- Deployment of Stage 1;
- Deployment of Stage 2;
- Deployment of Stage 3.

These stages are described below.

Ballistic trajectory before the first stage is fully deployed. We may apply Eqs.(6), which for the altitude range of $h \in [H_0 - \Delta H; H_0]$ yield

$$x(h) = x_0 + \int_{H_0-\Delta H}^{H_0} \frac{w(h)\cos\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{H_0 - h}{g}}V_0\cos\Psi_0$$
$$y(h) = y_0 + \int_{H_0-\Delta H}^{H_0} \frac{w(h)\sin\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{H_0 - h}{g}}V_0\sin\Psi_0 \tag{8}$$

Specifically, by the time (altitude) the first stage deploys we have

$$x_1 = x_0 + \int_{H_0-\Delta H}^{H_0} \frac{w(h)\cos\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{\Delta H}{g}}V_0\cos\Psi_0$$
$$y_1 = y_0 + \int_{H_0-\Delta H}^{H_0} \frac{w(h)\sin\psi_w(h)}{\sqrt{2g\left(H_0 - h\right)}}\,dh + \sqrt{2\frac{\Delta H}{g}}V_0\sin\Psi_0 \tag{9}$$

From this point on we assume that the initial horizontal speed caused by deploying from an aircraft flying with speed V_0 is phased out completely. For simplicity we will not consider the gradual transition from a ballistic fall to the normal gliding descent. Indeed, accounting for this transition may cause some minor changes in computation of the safety fan but being on the conservative side, we will not care that the size of the fan might be smaller without the transition. In any case, it is likely that wind changes may introduce lager error than neglect of the transition.

Descent of a fully deployed system. After canopy deployment the trajectory obeys the following set of ordinary differential equations:

$$\begin{aligned}\frac{dx}{dt} &= w(h)\cos\psi_w(h) + v_{v1}GR_1\cos\psi \\ \frac{dy}{dt} &= w(h)\sin\psi_w(h) + v_{v1}GR_1\sin\psi \\ \frac{dh}{dt} &= v_{v1}\end{aligned} \tag{10}$$

Here we assume that the descent rate does not depend on the altitude (air density), and is constant during the entire decent. Using the third equation as demonstrated previously, the first two equations can be reduced to

$$\begin{aligned}\frac{dx}{dh} &= v_{v1}^{-1}w(h)\cos\psi_w(h) + GR_1\cos\psi \\ \frac{dy}{dh} &= v_{v1}^{-1}w(h)\sin\psi_w(h) + GR_1\sin\psi\end{aligned} \tag{11}$$

Now, we can integrate these equations with respect to an altitude, so that

$$\begin{aligned}x(h) &= x_1 + \int_h^{H_0-\Delta H}\left(v_{v1}^{-1}w(h)\cos\psi_w(h) + GR_1\cos\psi\right)dh \\ y(h) &= y_1 + \int_h^{H_0-\Delta H}\left(v_{v1}^{-1}w(h)\sin\psi_w(h) + GR_1\sin\psi\right)dh\end{aligned} \tag{12}$$

Again, the integrals can be split into two parts

$$\begin{aligned}x(h) &= x_1 + \int_h^{H_0-\Delta H} v_{v1}^{-1}w(h)\cos\psi_w(h)dh + (H_0-\Delta H-h)GR_1\cos\psi \\ y(h) &= y_1 + \int_h^{H_0-\Delta H} v_{v1}^{-1}w(h)\sin\psi_w(h)dh + (H_0-\Delta H-h)GR_1\sin\psi\end{aligned} \tag{13}$$

where the third terms represent the contribution from the stuck controls and the second terms provide the contribution of the variable winds. By the altitude of deployment of the second stage we will have

$$\begin{aligned}x_2 &= x_1 + \int_{H_2}^{H_0-\Delta H} v_{v1}^{-1}w(h)\cos\psi_w(h)dh + (H_0-\Delta H-H_2)GR_1\cos\psi \\ y_2 &= y_1 + \int_{H_2}^{H_0-\Delta H} v_{v1}^{-1}w(h)\sin\psi_w(h)dh + (H_0-\Delta H-H_2)GR_1\sin\psi\end{aligned} \tag{14}$$

Deployment of Stage 2. If Stage 2 is present then the equations of motion will be similar to those of Eqs.(12)-(14). Specifically, we can write

$$\begin{aligned}x(h) &= x_2 + \int_h^{H_2} v_{v1}^{-1}w(h)\cos\psi_w(h)dh + (H_2-h)GR_2\cos\psi \\ y(h) &= y_2 + \int_h^{H_2} v_{v1}^{-1}w(h)\sin\psi_w(h)dh + (H_2-h)GR_2\sin\psi\end{aligned} \tag{15}$$

and

$$\begin{aligned}x_3 &= x_2 + \int_{H_3}^{H_2} v_{v1}^{-1}w(h)\cos\psi_w(h)dh + (H_2-H_3)GR_2\cos\psi \\ y_3 &= y_2 + \int_{H_3}^{H_2} v_{v1}^{-1}w(h)\sin\psi_w(h)dh + (H_2-H_3)GR_2\sin\psi\end{aligned} \tag{16}$$

respectively.

Deployment of Stage 3. Finally, for Stage 3 (if present) we will have

$$x(h) = x_3 + \int_h^{H_3} v_{v1}^{-1} w(h) \cos\psi_w(h) dh + (H_3 - h) GR_3 \cos\psi$$
$$y(h) = y_3 + \int_h^{H_3} v_{v1}^{-1} w(h) \sin\psi_w(h) dh + (H_3 - h) GR_3 \sin\psi \tag{17}$$

and

$$x(0) = x_3 + \int_0^{H_3} v_{v1}^{-1} w(h) \cos\psi_w(h) dh + H_3 GR_3 \cos\psi$$
$$y(0) = y_3 + \int_0^{H_3} v_{v1}^{-1} w(h) \sin\psi_w(h) dh + H_3 GR_3 \sin\psi \tag{18}$$

Now that we have developed the equations of motion for a multi-stage system, let us consider a representative example.

III. Example of Safety Fans for a Two-Stage System

Consider a system that has two stages, which are described as follows:

	v_i, m/s	GR_i	ΔH (H_2), m
Stage 1	18	2.5	666
Stage 2	6	0	333

Assume we release it from $H_0 = 1,500m$ at $V_0 = 60m/s$ due North-West. The winds are given in a look-up table and stored in the `Winds(:,1:3)` matrix (where the first column contains altitude, second – wind direction, and third – wind magnitude). (Hereafter we will use fragments of self-explanatory actual code written in MATLAB.)

Depending on the data collection method the wind measurements may be not distributed evenly in altitude. So, before computing definite integrals in Eqs.(6),(8),(12),(15), or (17) numerically (using trapezoidal numerical integration) the original look-up table h_k, w_k, ψ_{wk} should be approximated with, for example, piecewise cubic Hermite interpolating polynomials $w(h)$ and $\psi_w(h)$:

```
%% Setting initial conditions
V0=60; H0=1500; Psi0=45*pi/180; g=9.81; DR=[18; 6]; GR=[2.5 0];
DH=666; H2=333;
h=linspace(0,H0,98);
h(end+1)=H0-DH; h(end+1)=H2; h=sort(h);
%% Producing spline interpolations for the winds
psih = pchip(Winds(:,1),Winds(:,2),h)*pi/180;
wh   = pchip(Winds(:,1),Winds(:,3),h);
```

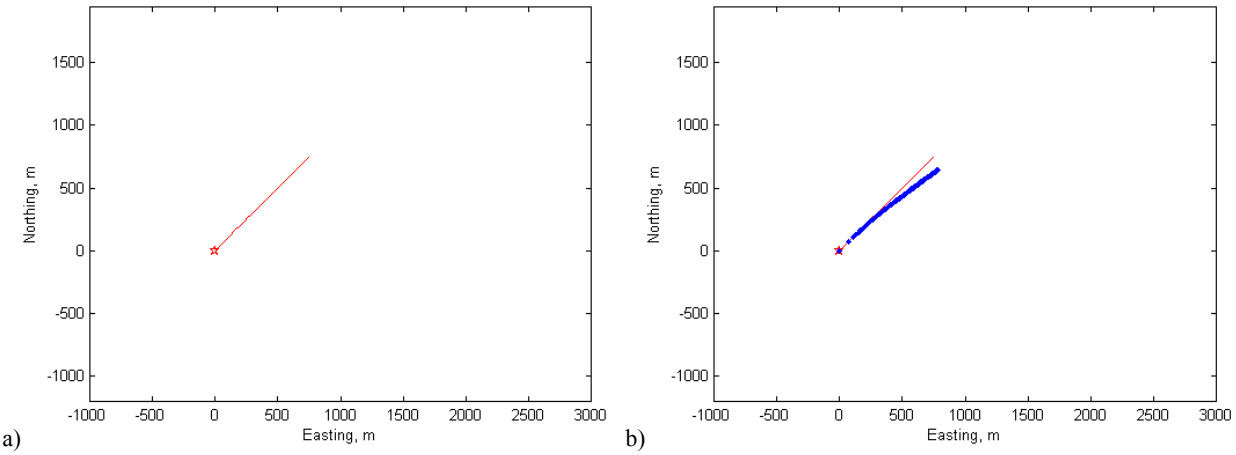
Next, the ballistic trajectory due to initial velocity (the last terms in Eqs.(6)) can be computed as follows:

```
%% Computing a ballistic fall if no winds were present
xt1=sqrt(2*(H0-h)/g)*V0*cos(Psi0);
yt1=sqrt(2*(H0-h)/g)*V0*sin(Psi0);
```

If accounting for the winds (the second terms in Eqs.(6)) the trajectory becomes

```
%% Computing contribution of winds
  for i=1:99
  xw(i)  = trapz(h(i:100),wh(i:100).*cos(psih(i:100))./sqrt(2*g*H0-h(i:100)));
  yw(i)  = trapz(h(i:100),wh(i:100).*sin(psih(i:100))./sqrt(2*g*H0-h(i:100)));
  end
xt2=xt1+[xw 0];
yt2=yt1+[yw 0];
```

Both trajectories are shown in Fig.1 (the star represents a release point). This latter trajectory represents the ballistic fall if the first- (and second-) stage canopy does not open. The winds are pushing the trajectory to the South-East but because the system falls ballistically (very quickly) the winds introduce no major change.



a) b)

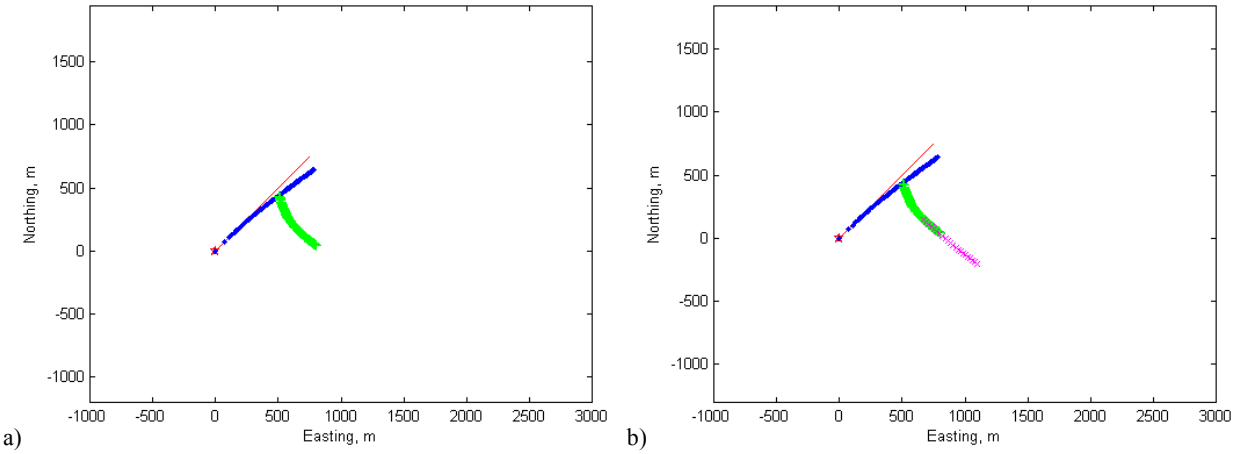
Fig.1 Ballistic trajectories with no wind (a) and winds (b).

Now, deploying the first stage canopy means that we are slowing down the descent and the winds will produce a much greater effect. Fig.2a shows the addition of the second terms in Eq.(14) computed as follows:

```
%% Deploying the 1st stage
index1=find(h>H0-DH); Ind1=index1(1)-1;
  for i=1:Ind1-1
  xS1(i)  = trapz(h(i:Ind1),wh(i:Ind1).*cos(psih(i:Ind1))./DR(1));
  yS1(i)  = trapz(h(i:Ind1),wh(i:Ind1).*sin(psih(i:Ind1))./DR(1));
  end
xS1 = xS1+xt2(Ind1); yS1 = yS1+yt2(Ind1);
```

The only difference from Eq.(14) is that the trajectory is propagated all way down to the surface rather than to H_2. When the second stage is deployed (which leads to even slower rate of descent) the winds play an even greater role (Fig.2b):

```
%% Deploying the 2nd stage
index2=find(h>H2); Ind2=index2(1)-1;
  for i=1:Ind2-1
  xS2(i)  = trapz(h(i:Ind2),wh(i:Ind2).*cos(psih(i:Ind2))./DR(2));
  yS2(i)  = trapz(h(i:Ind2),wh(i:Ind2).*sin(psih(i:Ind2))./DR(2));
  end
xS2 = xS2+xS1(Ind2); yS2 = yS2+yS1(Ind2);
```



a) b)

Fig.2 The first-stage opening (no glide) (a) and the second-stage opening (no glide) (b).

Finally, the effect of the controls failure mode (controls stuck in one position leading to a glide with constant ground track angle) is reduced to drawing a circle of appropriate radius (the last term in Eq.(14)) around the last point of the current stage.

```
%% Computing a safety fan for the first stage
[X,Y,Z] = cylinder((H0-DH-H2)*GR(1));
%% Computing a safety fan for the second stage
[X,Y,Z] = cylinder((H0-DH-H2)*GR(1)+H2*GR(2));
```

The results are presented in Fig.3. Since the specific system we are considering uses round parachutes with no glide as a second stage, the final safety fan is simply first-stage safety fan shifted South-East.

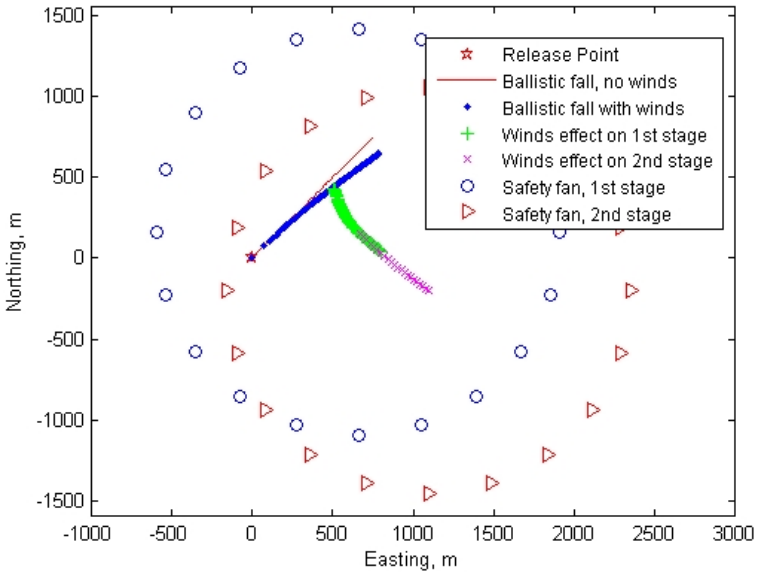


Fig.3 Safety fans in case the controls are stuck in a fixed position.

Computation of the safety fans for different multistage systems is carried out similarly.

As seen, the computations above are not very complicated and yet, they are unique for each system and carried out numerically based on the unevenly-spaced measurements of the winds aloft. If two different systems are to be dropped, these calculations should be performed for each system individually. The goal, however, is to have another technique allowing the use of some sort of normalized winds that can be computed upfront once and then be applied to any system. This brings us to the next section, which discusses the so-called ballistic winds.

IV. Computation of Ballistic Winds

For operational use the vertical winds profile can be reduced to the so-called ballistic winds. In other words, if at some altitude H we have a ballistic wind of magnitude W and direction Ψ_W, then the effect of variable winds for some system with descent rate V_v on its way from altitude H down to the surface is reduced to simple formulas:

$$\begin{aligned} x(h) &= \frac{H}{V_v} W \cos\Psi_W \\ y(h) &= \frac{H}{V_v} W \sin\Psi_W \end{aligned} \tag{19}$$

A. Deriving Ballistic Winds

Comparing Eqs.(19) with Eqs.(18) it becomes clear that

$$\begin{aligned} \int_0^H \frac{1}{V_v} w(h)\cos\psi_w(h)dh &= \frac{H}{V_v} W \cos\Psi_W \\ \int_0^H \frac{1}{V_v} w(h)\sin\psi_w(h)dh &= \frac{H}{V_v} W \sin\Psi_W \end{aligned} \tag{20}$$

Substituting integrals with the finite sum of trapezoids based on the discrete values of h_k, w_k and ψ_{wk}, $k = 1,...,M$, we get

$$\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\frac{w_k \cos\psi_{w;k} + w_{k-1}\cos\psi_{w;k-1}}{2} = h_M W_M \cos\Psi_{W;M}$$
$$\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\frac{w_k \sin\psi_{w;k} + w_{k-1}\sin\psi_{w;k-1}}{2} = h_M W_M \sin\Psi_{W;M} \tag{21}$$

The index starts from 2 because by definition the winds measurements at the lowest altitude can be considered ballistic winds at this altitude.

From Eqs.(21) it further follows that

$$\tan\Psi_{W;M} = \frac{\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\left(w_k \sin\psi_{w;k} + w_{k-1}\sin\psi_{w;k-1}\right)}{\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\left(w_k \cos\psi_{w;k} + w_{k-1}\cos\psi_{w;k-1}\right)}$$
$$W_M = \frac{1}{2h_M}\sqrt{\left(\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\left(w_k \cos\psi_{w;k} + w_{k-1}\cos\psi_{w;k-1}\right)\right)^2 + \left(\sum_{k=2}^{M}\left(h_k - h_{k-1}\right)\left(w_k \sin\psi_{w;k} + w_{k-1}\sin\psi_{w;k-1}\right)\right)^2} \tag{22}$$

For the specific case when $h_k - h_{k-1} = \Delta h = const$, $k = 2,...,M$, Eqs.(22) can be further reduced to

$$\tan\Psi_{W;M} = \frac{\sum_{k=2}^{M}\left(w_k \sin\psi_{w;k} + w_{k-1}\sin\psi_{w;k-1}\right)}{\sum_{k=2}^{M}\left(w_k \cos\psi_{w;k} + w_{k-1}\cos\psi_{w;k-1}\right)}$$
$$W_M = \frac{\Delta h}{2h_M}\sqrt{\left(\sum_{k=2}^{M}\left(w_k \cos\psi_{w;k} + w_{k-1}\cos\psi_{w;k-1}\right)\right)^2 + \left(\sum_{k=2}^{M}\left(w_k \sin\psi_{w;k} + w_{k-1}\sin\psi_{w;k-1}\right)\right)^2} \tag{23}$$

B. Example of Computing Ballistic Winds

Let us consider a simple example. Suppose that the winds aloft were measured as

Alt, ft	Direction, deg	Magnitude, kts
0	93	10
1000	101	36
2000	119	25
3000	129	21
4000	135	19
5000	139	17
6000	145	16
7000	150	16
8000	152	16
9000	151	16
10000	144	16

and stored into the `Winds` matrix. Then, the following script

```
BWinds(:,1)=Winds(:,1); BWinds(1,2)=Winds(1,2); BWinds(1,3)=Winds(1,3);
  for n=2:11
  dh=diff(Winds(1:n,1));
  wE=sum(dh.*(Winds(2:n,3).  *sin(Winds(2:n,2)  *pi/180)+...
              Winds(1:n-1,3).*sin(Winds(1:n-1,2)*pi/180)))/2;
  wN=sum(dh.*(Winds(2:n,3).  *cos(Winds(2:n,2)  *pi/180)+...
              Winds(1:n-1,3).*cos(Winds(1:n-1,2)*pi/180)))/2;
  BWinds(n,2)=atan2(wE,wN)*180/pi;
  BWinds(n,3)=sqrt(wE^2+wN^2)/Winds(n,1);
  end
```

converts them into the ballistic winds:

```
    0.0         93.0         10.0
 1000.0         99.3         23.0
 2000.0        104.4         26.5
 3000.0        110.2         25.0
```

```
 4000.0        114.7          23.5
 5000.0        118.2          22.1
 6000.0        121.2          21.0
 7000.0        124.1          20.1
 8000.0        126.8          19.4
 9000.0        129.1          18.8
10000.0        130.6          18.5
```

Now, if for instance we want to see what would happen to the second stage of the system described in Section III if it were deployed at 3,000ft, then instead of calculating integrals in Eq.(18) (which would result in a trajectory shown in Fig.4) we could simply apply Eq.(19):

```
X=BWinds(3,1)/GR(2)*BWinds(3,3)*cos(BWinds(3,2)*pi/180);
Y=BWinds(3,1)/GR(2)*BWinds(3,3)*sin(BWinds(3,2)*pi/180);
```

This accurately computes the forecast point of impact (shown in Fig.4 as a solid circle) in one shot.

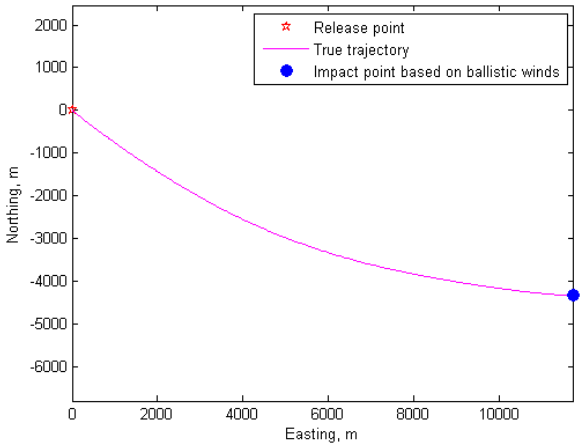


Fig.4 Calculation of a predicated impact point based on ballistic winds.

To summarize, the ballistic winds (when computed correctly according to Eqs.(22) and (23)) provide an excellent tool to rapidly assess the impact point in case of canopy opening failure and descent of a symmetric canopy ADS (with zero glide ratio). However, the descent of a parafoil ADS still requires rigorous computations. As a result, the next section presents a GUI developed to perform these computations behind the scene, while visualizing the drop planning and all safety related information on one screen. This provides the capability to conveniently change / adjust / play with different parameters.

V. Interactive GUI

Figure 5 presents a screen shot of the developed GUI which performs various computations and manipulations (the factual data is replaced with fake data). This GUI was developed in the MATLAB development environment using the Open GUI Layout Editor GUIDE. The code that runs this GUI contains over 3000 lines and the GUI itself is subdivided into several panels, described next.

A. Winds Panel

The Winds panel allows the user to browse for wind files and chose the most appropriate one. The wind source files are sounding, Windpack, balloon and JAAWIN winds. Examples of heading and the first three lines of wind data for each of these files are shown in Figs.6-9. To read each of these files a corresponding function was created. For instance, the following set of commands

```
%% Reading heading information
[Y,M,D,H,f1,f2,f3,lat,lon]=...
  textread(FileName,'%4c-%2c-%2c_%2c %s i,j=( %f , %f ),lat,long=( %f , %f',...
           1,'headerlines',2);
Ldate=[M '/' D '/' Y];, Ltime=[H ':00'];
file = textread(FileName, '%s', 'delimiter', '\n');
%% Reading numerical data
```

```
N=length(file); i=6; Winds=[0 0 0];
while i <= N
[q,w,e,r,t,y,u]=textread(FileName,'%f %f %f %f %f %f %f',1,'headerlines',i);
Winds=vertcat(Winds,[w y t]); % the format is altitude, Wdir, Wspeed
i=i+1;
end
Winds(1,:)=[];
```

reads data from the sounding file (Fig.6).

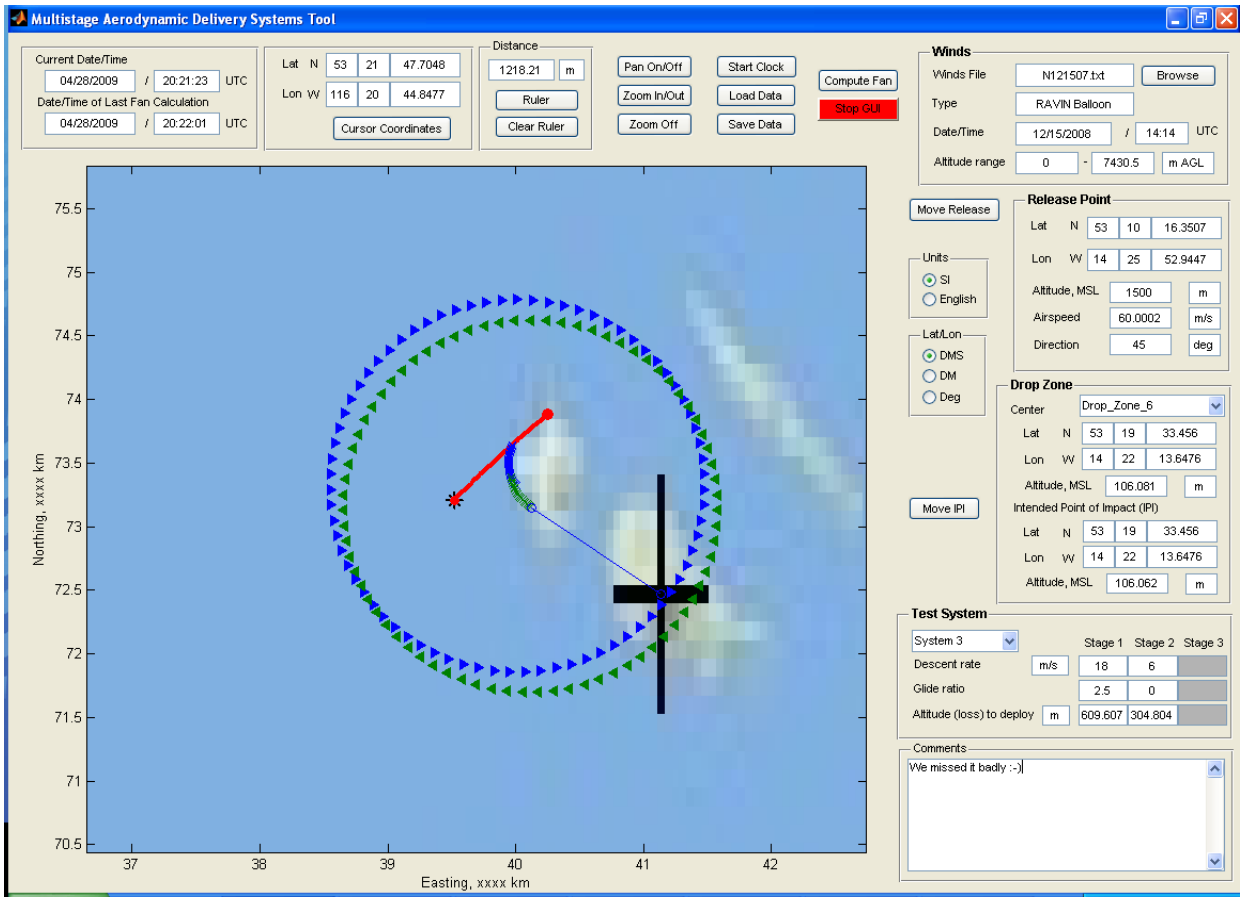


Fig. 5 Safety fans GUI.

```
37    2
2008-12-16_00Z,DZID00         i,j=( 15.9,  15.6),lat,long=(54.832,83.104)
 SFALT,    SFPRES
  381     968.3
  LINE   AGL(m)      T(C)     RH(%)    WSPD(m/s)   WDD     P(mb)
    1       2.0     11.76     50.93      9.33    182.93    968.04
    2      15.2     11.64     51.16     10.97    182.91    966.53
    3      56.2     11.26     52.30     13.47    182.37    961.80
```

Fig. 6 Example of a winds sounding file.

```
Time(UTC)    Lat(deg)    Long(deg) Altitude(M-HAE) Q StdDev VE(M/sec) VN(M/Sec) VUp(M/Sec)
53728.000    54.832114   83.104813    5613.796     4 0.697   -0.970    91.957     -0.030
53728.100    54.832114   83.104813    5613.817     4 0.697   -0.967    91.811      0.142
53728.200    54.832114   83.104813    5613.828     4 0.697   -1.097    91.665      0.102
```

Fig. 7 Example of a Windpack winds file.

0700L FLIGHT

Station Name: DZ00 Radiosonde Type: RS92-SGP
Latitude: 54.83 Radiosonde Number: DXXXXXX
Longitude: 83.10 Altitude: 1322 feet (403 Meters)
Ground Check Corrections
Pressure: 0.44 mb RH1: 1.07 %
Temperature: -0.84 °C RH2: -0.41 %
Launch Time: 14:14 UTC Sounding Started on: 12/15/2008

Altitude (FT above msl)	Press (MB)	Temp (°F)	RH (%)	Dew Pt Temp (°F)	Air Density (g/m^3)	Wind Direct (degs)	Wind Speed (Kts)
1322	968.0	45.6	44	24.9	1197.94	124.0	9.0
1400	965.2	45.4	41	23.0	1195.07	127.0	8.9
1500	961.6	45.5	41	23.3	1190.34	129.5	10.7

Fig. 8 Example of a balloon winds file.

A. Forecast Time, Drop Zone Location, Elevation
Valid Time: 1400Z 27 Oct 2008 Data Source: MM5
Forecast Model Grid Pt: Latitude-> 54.83N Longitude-> 83.104E
Forecast Model Grid Pt Elevation: 1719 feet above MSL
Point of Impact: Latitude-> 54.83N Longitude-> 83.104W
Forecast Model Grid Point Bearing/Range from Point of Impact: 90DEG/0.00KM
(*Note! Model Grid Pt is nearest to entered Lat/Lon and is in decimal degrees format.)
B. Winds, Temperature, Pressure Altitude, D-Values, Ballistic Winds -

HEIGHT(PRESS) (KFT AGL)(MB)	DIR (DEG)	SPEED (KNOTS)	TEMP (DEG C)	PRESS ALT (KFT MSL)	D-VALUE (FT)	BALLISTIC WINDS (DEG)	(KNOTS)
SFC(959)	93	10	20	1.61	+0198	93	10
1 (926)	101	36	22	2.47	+0245	96	18
2 (894)	119	25	25	3.42	+0294	108	18
3 (863)	129	21	24	4.36	+0355	108	18

Fig. 9 Example of a JAAWIN winds forecast file.

The program will automatically recognize what type of file was chosen and how to read it properly. The wind data will then be loaded into the appropriate winds matrix. Auxiliary data such as the name and type of the file, when and where the winds data were taken, and altitude range will also be read and displayed in the corresponding windows.

B. Drop Zone and Map Panels

The Drop Zone panel allows the user to choose one of the drop zones stored in an ASCII file. The script behind this panel loads latitude, longitude and mean-sea-level altitude of a chosen drop zone. The corresponding map appears on the left. By default the center of the chosen DZ becomes an intended impact point (shown in the corresponding windows). It can be changed by selecting a new point on the map or typing new values in the windows.

The map can be any graphical image scaled into a rectangle with respect to the Universal Transverse Mercator (UTM) coordinates. When the map loads, the axes of this map are also assigned so that they can be manipulated later. For instance, the following set of commands loads the jpg image of some map:

```
A = imread('MAP_UTM_WGS84','jpg');
MAPimageR = A(:,:,1:3); MAPimage(:,:,1:3) = MAPimageR(end:-1:1,:,1:3);
clear MAPimageR A
    Mlat=[60.17 123.51];
    Mlon=[53.65  63.93];
image(Mlon, Mlat, MAPimage); axis([Mlon Mlat])
set(gca,'YDir','normal'); axis equal
```

The user can zoom the map in and out, move it, measure distances between selected points, get the coordinates of any point selected with the cursor, and read the coordinates of any point in one of the chosen formats (UTM on the map, or Lat/Lon in the windows). Two functions support conversion of UTM coordinates into Lat/Lon and back.

C. Test System Panel

The Test System panel allows the user to choose one of the ADSs stored in an Excel file (see Fig.10). This system can have one, two, or three stages. The corresponding windows for unused stages are shadowed. The only ADS data needed for fan computation are the system's descent rate, glide ratio and altitude to deploy (altitude loss for the first stage). The user is allowed to change any of these three parameters in the corresponding windows or create an entirely new system. This data along with winds aloft and release point information is used to compute safety fans.

		Stage 1			Stage 2		
System ID	Number of stages	Descent rate, m/s	Glide Ratio	Altitude Loss to Deploy, ft	Descent rate, m/s	Glide Ratio	Altitude to Deploy, ft
New System	1	4.5	3	2000			
System 1	1	8	0.6	2000			
System 2	1	8	0.6	2000			
System 3	2	18	2.5	2000	6	0	1000
System 4	2	18	2.5	2000	6	0	1000
System 5	1	4.5	2.5	2000			
System 6	1	5.5	2.5	2000			
System 7	1	5.25	3.5	2500			
System 8	1	4.88	3.5	2500			
System 9	1	4.88	3.1	1500			
System 10	1	5.25	3.5	2000			
System 11	1	4.75	3.5	1000			
System 12	1	4	4	1000			
System 13	1	9	3	1000			
System 14	2	8	3	1000	5	0	1000
System 15	1	4.8	3.5	2000			
System 16	1	4.8	3.5	2000			
System 17	1	4.9	3.1	1000			
System 18	1	4.95	3.1	2000			
System 19	1	4.9	3.1	400			
System 20	1	4.2	3.2	400			
System 21	1	4.4	3.5	2000			
System 22	1	3.66	2	400			
System 23	1	8.53	0	2000			
System 24	1	22	0	2000			
System 25	1	8.53	0	2000			
System 26	1	22	0	2000			
System 27	2	58	0	0	7	0	3300

Fig. 10 Examples of ADS input data.

D. Release Point and Units Panels

The idea behind the Release Point panel is that based on the system being tested and available winds data, the user enters various release points and computes safety fans. If the safety requirements are not met, the user moves the release point and a new safety fan can be generated. The release point can be moved by clicking on a new location in the map, in which case the new coordinates immediately appear in the corresponding panel windows, or by entering a new latitude and longitude in the panel windows. Altitude, airspeed, and direction at the release point are provided by the user. When a satisfactory release point has been chosen, its coordinates will be provided to the crew for execution.

As an example, the following commands are activated upon clicking the "Move Release" button in the GUI and choosing to select a new point by clicking on the map. The current release point, if any exists, is deleted from the map. The `ginput()` function is interactive and allows the user to click anywhere in the map to select a point. The new release point is then plotted and the corresponding Lat/Lon calculated:

```
%% Deleting current release point if any
h = findobj('MarkerFaceColor','k','-and','MarkerSize',12);
h2 = findobj('MarkerFaceColor','r','-and','MarkerSize',6);
delete(h); delete(h2);
%% Getting and plotting chosen release point
[x,y] = ginput(1);
rp = plot(x,y,'k*');
set(rp,'MarkerFaceColor','k','MarkerSize',12,'LineWidth',2);
rp2 = plot(x,y,'ro');
set(rp2,'MarkerFaceColor','r','MarkerSize',6);
%% Determining lat/lon
```

```
easting = x*10^3 + 350000;
northing = y*10^3 + 1800000;
zone = '11S';
[RPlatDeg,RPlonDeg]=utm2wgs(easting,northing,zone);
%% Accounting for sign of longitude (using West in GUI vs. negative)
RPlonDeg = -RPlonDeg;
```

Since the ground crew and the deployment crew can speak different technical languages (units), the Units panel allows the user to change units from the International System of Units (SI) to English and back. In addition, Lat/Lon coordinates can be visualized throughout all panels in one of the following three formats:

- Deg – degrees.decimal degrees
- DM – degrees, minutes.decimal minutes
- DMS – degrees, minutes, seconds.decimal seconds

If using the DM or Deg formats, the unused windows are shadowed. For example, the chosen format in Fig.5 is DM, and therefore the first windows in all Lat/Lon readings are shadowed.

E. Operations

Other operational functions in the GUI include timing, fan computation, loading and saving data, and commenting. The "Start Clock" button asks the user for the computer's time zone setting and starts a continuously updating clock in Coordinated Universal Time (UTC). When all the necessary data for fan computation has been selected and entered, the user can compute the current safety fan. Corresponding graphics appear on the map. The time of the most recent calculation appears under the running clock.

The GUI can be saved with the Save Data button. Once a GUI has been saved, it can be loaded with the Load Data button. When reloaded, the GUI is still interactive. This means that though it can be used as a simple figure capturing one state in time, it can also be used as the starting point for continuing operations. If using a loaded GUI, the clock must be restarted but all other data are available.

The user can enter comments on a particular configuration in the Comments panel. These comments will be saved with the GUI. The Stop GUI button stops all activity in the figure and is used just prior to closing the application.

VI. Conclusions

Computation of the safety fans for testing of aero delivery systems involves simple and yet cumbersome computations. For uncontrolled systems the precomputed ballistic winds can reduce some of the computational burden. However, for effective operations with any system, especially the ones with several stages, some kind of automation is required. This paper presented an example of such an automated tool which performs several safety checks in convenient graphical form. The mathematical foundation for safety fan computations was also rigorously developed. The graphical user interface is ready and will be used soon in a field environment. Feedback will be provided to the developers for further improvement and modifications.

Acknowledgments

The authors would like to thank the team of the Air Delivery Systems Branch of the U.S. Army YPG for funding this project and providing some real test data to verify computational algorithms and the theory behind them.

References

1. *The Universal Grids: Universal Transverse Mercator (UTM) and Universal Polar Stereographic (UPG)*, DMA Technical Manual DMATM8358.2, Defense Mapping Agency, Faitfax, VA, 1989.